CPSC 468 Midterm Review

Chapter 1

0.1 (Turing Machine)**.** *A k-tape Turing Machine M is described by a tuple (Γ, Q, δ). Assume $k \geq 2$, with 1 read-only input tape and $k-1$ work tapes. The last work tape is assumed to be the output tape.*

- *Γ the finite alphabet of symbols that M may have on its tapes. Assume that Γ contains at least $\{0, 1, \square, \triangleright\}$.*
- *Q a finite set of possible states M's state register may be in. Assume that Q contains a q_{start} and q_{halt}.*
- *$\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^{k-1} \times \{L, S, R\}^k$ a transition function for M that takes in the current state and each head's read, and outputs the next state, with $k-1$ writes on all the work tapes, and movement direction for all k tapes.*

0.2 (Computing a Function)**.** *Let $f : \{0,1\}^* \rightarrow \{0,1\}^*$ and $T : \mathbb{N} \rightarrow \mathbb{N}$, with M a TM. We say that M computes f if for every $x \in \{0,1\}^*$, if M is initialized to the start configuration on input x, then it halts with $f(x)$ on the output tape. We say M computes f in $T(n)$-time if its computation on every x requires at most $T(|x|)$ steps.*

0.3 (Time Constructible)**.** *A function $T : \mathbb{N} \rightarrow \mathbb{N}$ is time constructible if $T(n) \geq n$ and there is a TM M that computes the function $x \mapsto \llcorner T(|x|) \lrcorner$ in time $T(n)$. $T(n) \geq n$ is to allow the algorithm to read its input. Some time constructible functions are n, $n \log n$, n^2 and 2^n.*

0.4. *For every $f : \{0,1\}^* \rightarrow \{0,1\}$ and time constructible $T : \mathbb{N} \rightarrow \mathbb{N}$, if f is computable in time $T(n)$ by some TM M using alphabet Γ, then it is able to compute the same function using $\{0, 1, \square, \triangleright\}$ in $(c \log_2 |\Gamma|) \cdot T(n)$. This is because we may express each symbol of Γ using $\log |\Gamma|$ binary bits, with some constant c overhead.*

0.5. *A k-tape TM can have its $k-1$ work tapes simulated by a single tape by interleaving the k tapes together.*

0.6 (Oblivious Turing Machine)**.** *An oblivious TM's head movement depends on the length of the input, not the contents of the input. Every TM can be simulated by an oblivious TM.*

0.7 (Turing Machine Representation)**.** *Every binary string $x \in \{0,1\}^*$ represents some TM, and every TM is represented by infinite such strings (think: comments in a language). The machine represented by x is denoted M_x.*

0.8 (Universal Turing Machine)**.** *There exists a TM $\mathcal{U}$ such that for every $x, \alpha \in \{0,1\}^*$, $\mathcal{U}(x, a) = M_\alpha(x)$, where M_α denotes the TM represented by α. Moreover, if M_α halts on input x within T steps, then $\mathcal{U}_\alpha(x)$ halts within $CT \log T$ steps, where C is a number independent of $|x|$, and depends only on M_α's alphabet size, number of tapes, and number of states. The cost of simulating any machine M_α has a logarithmic overhead, due to the alphabet size difference between M_α and $\mathcal{U}$. As $\mathcal{U}$ has a single tape, we do the trick over interleaving M_α's work tapes together.*

0.9 (Uncomputable Function)**.** *Define U as follows: for every $\alpha \in \{0,1\}^*$, if the machine defined by α accepts itself, such that $M_\alpha(\alpha) = 1$, then $U(\alpha) = 0$. In other words $U(\alpha) = 1 - M_\alpha(\alpha)$. There is no such TM that can compute U, because U will always negate it.*

0.10 (Halting Problem)**.** *A TM H such that $H(\alpha, x) = 1$ if $M_\alpha(x)$ halts, and yields 0 otherwise, does not exist. We can construct a wrapper TM W that invokes H on itself, and performs the opposite. Diagonalization motherfuckers!*

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

Examples